

輾轉相除法 (Euclidean Algorithm)

30	18
----	----

輾轉相除法 (Euclidean Algorithm)

1	30	18
---	----	----

輾轉相除法 (Euclidean Algorithm)

1	30	18
	18	

輾轉相除法 (Euclidean Algorithm)

1	30	18
	18	
	12	

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18		
	12		

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18	12	
	12		

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18	12	
	12	6	

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18	12	
2	12	6	

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18	12	
2	12	6	
	12		

輾轉相除法 (Euclidean Algorithm)

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

Greatest common divisor

```
if( di vi sor > di vi dend )  
{  
    di vi sor = di vi dend;  
    di vi dend = di vi sor;  
}
```

di vi sor	30	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
di vi dend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
if( di vi sor > di vi dend )  
{  
    di vi sor = di vi dend;  
    di vi dend = di vi sor;  
}
```

di vi sor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
di vi dend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
if( divisor > dividend )  
{  
    buffer = divisor;  
    divisor = dividend;  
    dividend = buffer;  
}
```

buffer		0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	30	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
if( divisor > dividend )  
{  
    buffer = divisor;  
    divisor = dividend;  
    dividend = buffer;  
}
```

buffer	30	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	30	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
if( divisor > dividend )  
{  
    buffer = divisor;  
    divisor = dividend;  
    dividend = buffer;  
}
```

buffer	30	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
if( divisor > dividend )  
{  
    buffer = divisor;  
    divisor = dividend;  
    dividend = buffer;  
}
```

buffer	30	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	30	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf [0]		00033AF0
buf [1]		00033AF4
buf [2]		00033AF8
buf [3]		00033AFC
buf [4]		00033B00
buf [5]		00033B04
buf [6]		00033B08
⋮		⋮

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]		00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf [0]	30	00033AF0
buf [1]	18	00033AF4
buf [2]	12	00033AF8
buf [3]		00033AFC
buf [4]		00033B00
buf [5]		00033B04
buf [6]		00033B08
⋮		⋮

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]		00033AF0
buf[1]		00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];
cout << buf[ 3 ] << endl;
```

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; i != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;
```

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
buf[ 2 ] = buf[ 0 ] % buf[ 1 ];  
buf[ 3 ] = buf[ 1 ] % buf[ 2 ];  
buf[ 4 ] = buf[ 2 ] % buf[ 3 ];  
cout << buf[ 3 ] << endl;
```

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
for( int i{ 0 }; buf[          ] != 0; i++ )  
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];  
cout << buf[    ] << endl << endl;
```

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[      ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[      ] << endl << endl;
```

i		00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]		00033AF0
buf[1]		00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;

```

i		00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[      ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[      ] << endl << endl;

```

i	0	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;
```

i	0	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;

```

i	1	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;

```

i	1	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;
```

i	2	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;

```

i	2	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[          ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[    ] << endl << endl;

```

i	3	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i		00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]		00033AF0
buf[1]		00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )  
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];  
cout << buf[ i ] << endl << endl;
```

i		00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i	0	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]		00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i	0	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i	1	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]		00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i	1	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;
```

i	2	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]		00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```

buf[ 0 ] = di vi dend;
buf[ 1 ] = di vi sor;
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];
cout << buf[ i ] << endl << endl;

```

i	2	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

Greatest common divisor

```
buf[ 0 ] = di vi dend;  
buf[ 1 ] = di vi sor;  
for( int i{ 0 }; buf[ i + 1 ] != 0; i++ )  
    buf[ i + 2 ] = buf[ i ] % buf[ i + 1 ];  
cout << buf[ i ] << endl << endl;
```

i	3	00033AE4
di vi sor	18	00033AE8
di vi dend	30	00033AEC
buf[0]	30	00033AF0
buf[1]	18	00033AF4
buf[2]	12	00033AF8
buf[3]	6	00033AFC
buf[4]	0	00033B00
buf[5]		00033B04
buf[6]		00033B08
⋮		⋮

```

int main()
{
    int dividend, divisor;
    int hold;
    cin >> dividend >> divisor;

    if( divisor > dividend )
    {
        hold = divisor;
        divisor = dividend;
        dividend = hold;
    }

    int buffer[ 258340 ];
    buffer[ 0 ] = dividend;
    buffer[ 1 ] = divisor;

    for( int i{ 0 }; buffer[ i + 1 ] != 0; i++ )
        buffer[ i + 2 ] = buffer[ i ] % buffer[ i + 1 ];

    cout << buffer[ i ] << endl << endl;
}

```


Don't use arrays

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

divisor

dividend

0012FF74

0012FF75

0012FF76

0012FF77

0012FF78

0012FF79

0012FF7A

0012FF7B

0012FF7C

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

12

0012FF74

0012FF75

0012FF76

0012FF77

0012FF78

divisor

18

0012FF79

0012FF7A

0012FF7B

0012FF7C

dividend

30

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

12

0012FF74

0012FF75

0012FF76

0012FF77

divisor

18

0012FF78

0012FF79

0012FF7A

0012FF7B

dividend

18

0012FF7C

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

12

0012FF74

0012FF75

0012FF76

0012FF77

divisor

12

0012FF78

0012FF79

0012FF7A

0012FF7B

dividend

18

0012FF7C

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

6

0012FF74

0012FF75

0012FF76

0012FF77

0012FF78

divisor

12

0012FF79

0012FF7A

0012FF7B

0012FF7C

dividend

18

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

6

0012FF74

0012FF75

0012FF76

0012FF77

divisor

12

0012FF78

0012FF79

0012FF7A

0012FF7B

dividend

12

0012FF7C

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

6

0012FF74

0012FF75

0012FF76

0012FF77

0012FF78

divisor

6

0012FF79

0012FF7A

0012FF7B

0012FF7C

dividend

12

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

1	30	18	1
	18	12	
2	12	6	
	12		
	0		

remainder

0

0012FF74

0012FF75

0012FF76

0012FF77

divisor

6

0012FF78

0012FF79

0012FF7A

0012FF7B

dividend

12

0012FF7C

0012FF7D

0012FF7E

0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder		0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	30	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	12	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	30	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	12	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	18	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	12	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	12	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	6	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	12	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	18	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	6	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	12	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	12	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	6	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	6	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	12	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

Greatest common divisor

```
remainder = dividend % divisor;
while( remainder != 0 )
{
    dividend = divisor;
    divisor = remainder;
    remainder = dividend % divisor;
}
```

remainder	0	0012FF74
		0012FF75
		0012FF76
		0012FF77
divisor	6	0012FF78
		0012FF79
		0012FF7A
		0012FF7B
dividend	12	0012FF7C
		0012FF7D
		0012FF7E
		0012FF7F

```
int main()
{
    int dividend, divisor, buffer;
    cin >> dividend >> divisor;
    if( divisor > dividend )
    {
        buffer = divisor;
        divisor = dividend;
        dividend = buffer;
    }

    int remainder = dividend % divisor;
    while( remainder != 0 )
    {
        dividend = divisor;
        divisor = remainder;
        remainder = dividend % divisor;
    }
    cout << divisor << endl << endl;
}
```